電機工程學系 / 指導老師:潘宗龍 / 學生:陳信佑、楊承恩

使用SDGS設計低通濾波器

組別編號:　　通訊

摘要

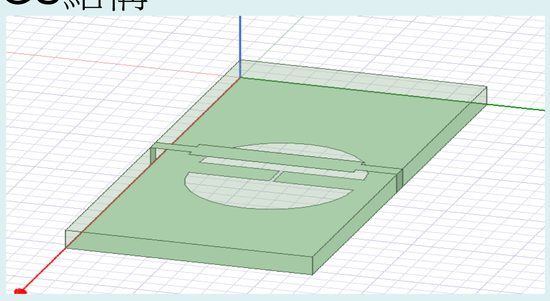
無線通訊快速發展使濾波器小型化與高性能需求增加。為提升微波電路效能，本研究採用半圓形地面缺陷結構(semicircle defected ground structure,SDGS)設計0.1－4 GHz 低通濾波器，並進行幾何與參數分析後並設計模型且製作樣品量測驗證。結果顯示量測與模擬高度一致，證明所提出的 SDGS 結構具良好濾波效果，可作為微波元件設計的重要參考。

研究方法

本研究使用半圓形地面缺陷結構，並利用模擬軟體HFSS進行模擬分析。而在研究方面則探討四種結構參數，缺陷圓形半徑、缺陷數量、訊號線均勻性和缺陷縫隙大小，並藉由四種特性的探討設計低通濾波器，再以FR-4材料製作樣品，並以網路分析儀測量，比對模擬與實際之數據。

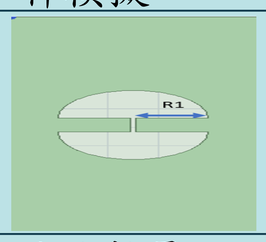
SDGS結構

在上層是訊號線，中間層是FR-4材料層，下層則是地面層，而在地面層挖了一個半圓形(類似於啞鈴)的缺陷。



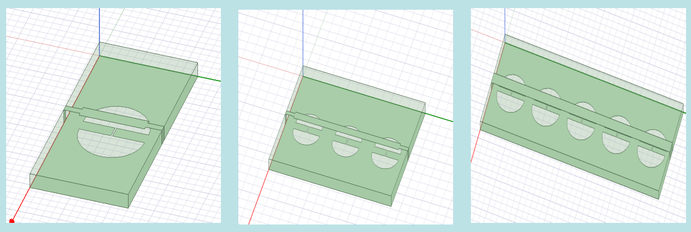
四種結構參數

1. 半圓形缺陷半徑
   對圓形缺陷半徑 R1 設定為 2 mm、3 mm及4 mm三種條件模擬



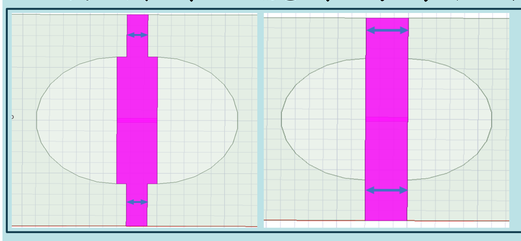
=>隨著缺陷半徑增加，截止頻率與共振頻率皆向低頻方向移動

2. 缺陷數量
   分別對圓形缺陷數量為1個、3個、5個三種條件模擬



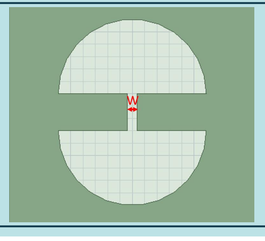
=>缺陷數量增加，截止頻率與共振頻率皆向低頻方向移動，且止帶衰減陡峭

3. 訊號線之均勻性
   分別對訊號線均勻和不均勻兩種條件模擬



=>越均勻反映出較穩定的低頻行為，有助於能量控制與濾波器性能預測

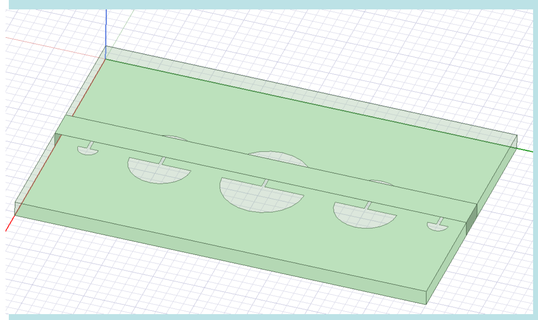
4. 半圓形之縫隙寬度
   分別對縫隙寬度 W設定為0.2 mm、0.3 mm及0.4 mm三種條件模擬



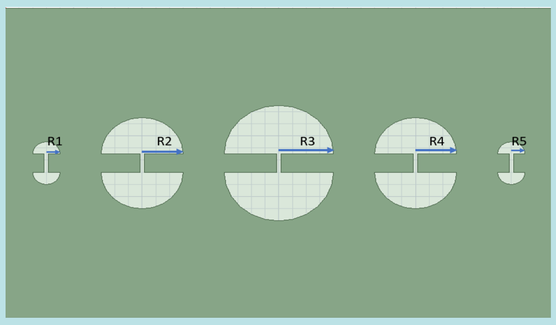
=>隨著縫隙寬度增加，截止頻率與共振頻率皆向高頻方向移動

濾波器架構

藉由四種特性設計一個訊號線寬度為3mm，地面層有五個缺陷，半徑分別是R1=1mm、R2=3mm、R3=4mm、R4=3mm、R5=1mm(圖六)，缺陷中間縫隙為0.3mm的低通濾波器(圖五)。



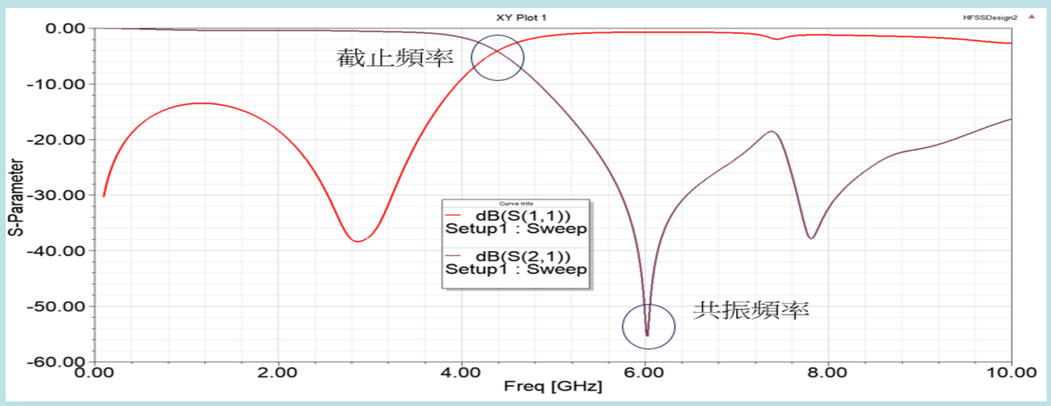
圖五 濾波器結構



圖六 缺陷半徑示意圖

結果與分析

從圖七和表一可知截止頻率在4.3GHz，而在共振頻率6.02GHz達到最佳的濾波效果，在這之間反射係數達到-55dB的差距，代表訊號的能量傳輸過去的量相當小。



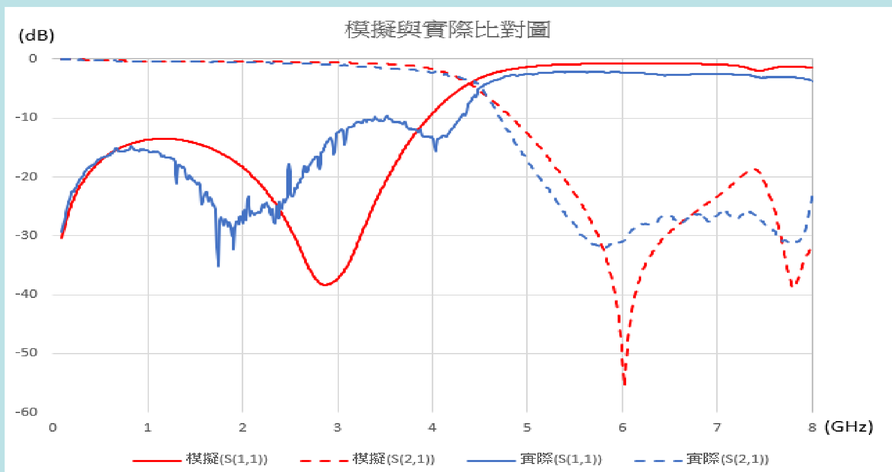
圖七 濾波器之頻率對反射系數之響應圖

Name	Freq[GHz]	S21[dB]
截止頻率	4.3941	-4.0650
共振頻率	6.0276	-55.3524

表一 截止頻率與共振頻率之頻率與反射係數

實際與模擬比較

從圖八可看出整體趨勢高度一致，有所差異主要來自製作過程中造成的微小尺寸誤差，使得實際樣品與模擬結果之間出現偏差。



圖八 模擬與實際比對圖

結論

本篇專題成功分析了半圓形地面缺陷結構的結構。研究結果發現，缺陷半徑增加或則是缺陷數量增加，會使濾波器的截止頻率與共振頻率皆向低頻移動，且缺陷數量增加還會使頻率響應在止帶衰減變陡峭；訊號線越均勻，其反映出較穩定的低頻反射行為；半圓形之縫隙寬度增加，截止頻率與共振頻率皆向高頻方向移動。所設計之低通濾波器效能優良，且濾波範圍接近在所需要的範圍0.1G-4GHz。經過了分析後，我們也更加了解這種結構的傳輸特性。

電機工程學系 / 指導老師:吳榮慶 / 學生:許万峻、蘇宏翊、江家凱

基於CNN與TRANSFORMER之太陽能光伏發電預測與NB-IOT即時介面系統開發

組別編號：　　能源

摘要

本專題於義守大學科技大樓搭建太陽能板數據蒐集與監控平台，並使用CNN-Transformer混和模型進行數據分析，以提高發電功率預測的準確性。系統透過RS-485與TTL模組建立穩定的通訊流程，且結合NB-IOT技術與MQTT協議，進行資料擷取與傳輸至資料庫。並以NODEMCU-32S架設網頁光伏發電數據監控平台，即時呈現太陽能板的發電量和設備狀況。數據經過清理與標準化處理後，輸入本專題之混合模型進行針對特定天氣特徵與時序分析提升預測精度。

研究方法

一、硬體設備架構

利用太陽能逆變器收集電壓、電流、頻率、功率及發電量等關鍵數據。系統採用RS-485通訊協定與低成本的TTL模組作為數據收發與自動控制裝置。核心控制採用ESP32微控制器，負責處理通訊協定，並透過ESP32內建的Wi-Fi模組與BC95的NB-IoT模組連接到路由器或基地台，實現與外部網路的穩定連線，支援低功耗、長距離的即時數據傳輸，可支

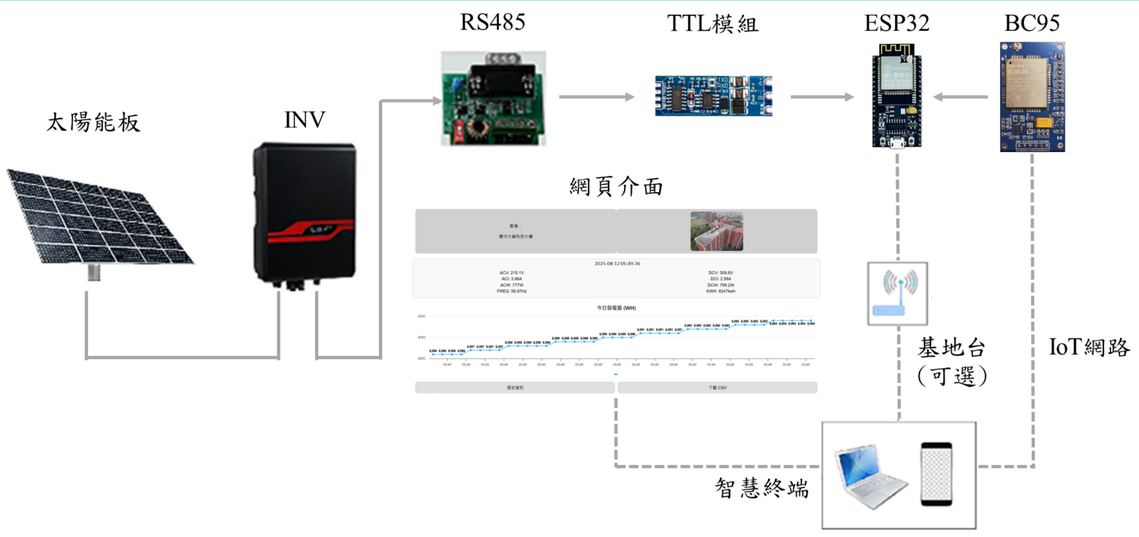


圖1、完整的即時監控系統結構圖

二、數據預處理

為提高數據的品質或減少其他因素的影響，需要對原始的數據進行預處理。

(1) 篩選與光伏發電最相關的資料，過濾無相關係數資料。

(2) 清洗缺失值和離群值。

(3) 消除物理維度的差異，加速模型的收斂速度， 對資料進行標準化。

(4) 將資料集劃分為訓練集、驗證集、測試集按照70%、15%、15% 劃分。

三、模型架構

該混合模型由CNN和自注意力機制的Transformer神經網絡構成。模型當中的CNN模型，被主要應用在對光伏功率發電的相關資料進行特徵的提取，為減小功率急劇波動時段預測的不確定性創造有利條件；而基於自注意力機制的Transformer神經網絡建立特徵間的時序依賴關係，同時也可以透過自注意力機制將資料當中重要對特定天氣氣候資料進行編碼並查看中相關資料的光伏發電功率數據，以處理天氣變化引起的功率波動問題。

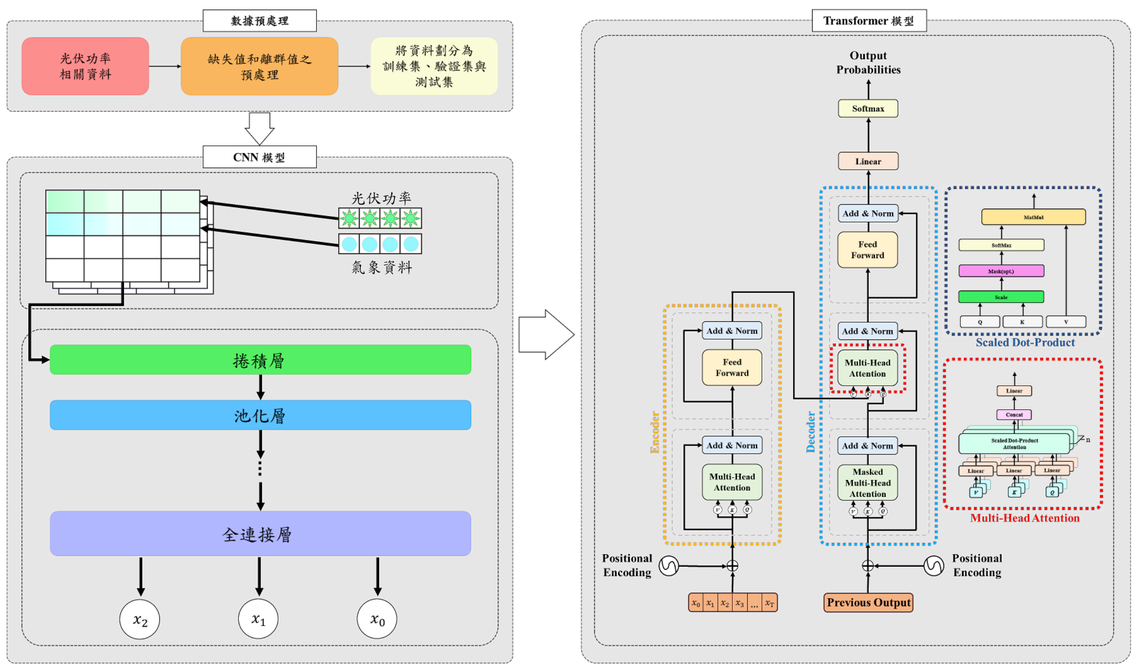


圖2、CNN-Transformer模型結構

結論

圖3、即時介面畫面(左：電腦端、右：手機端)

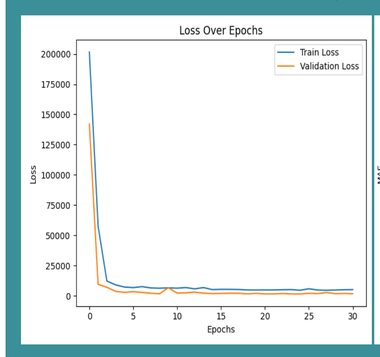


圖4、Loss

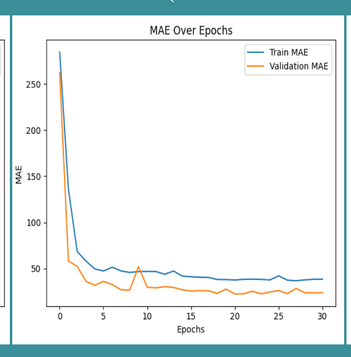


圖5、MAE

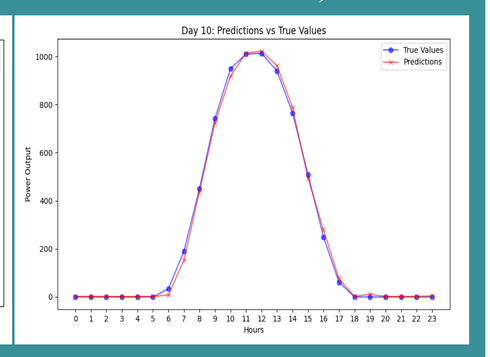


圖6、某日預測曲線圖

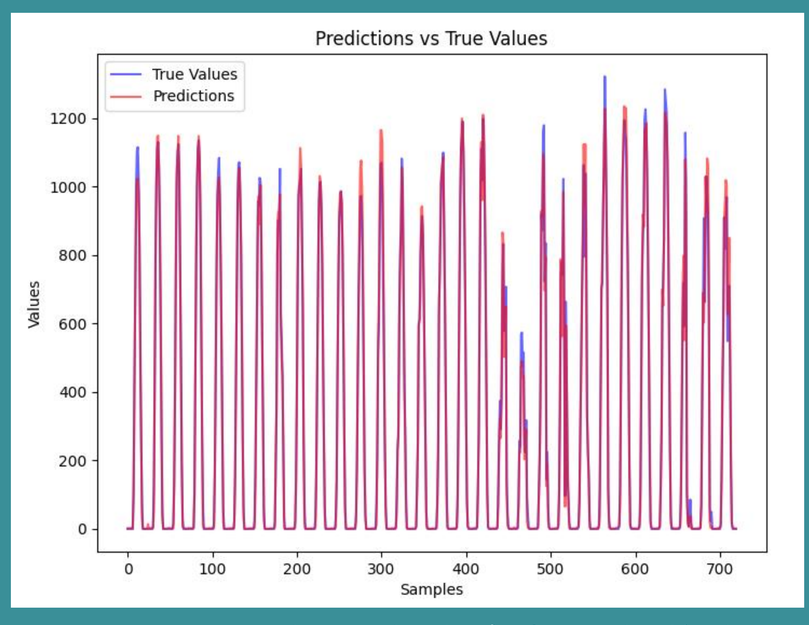


圖8、預測曲線(紅線)與真實曲線(藍線)

電機工程學系 / 指導老師:柯春旭 / 學生:王品方、周桓緯、黎彥慶、許家齊

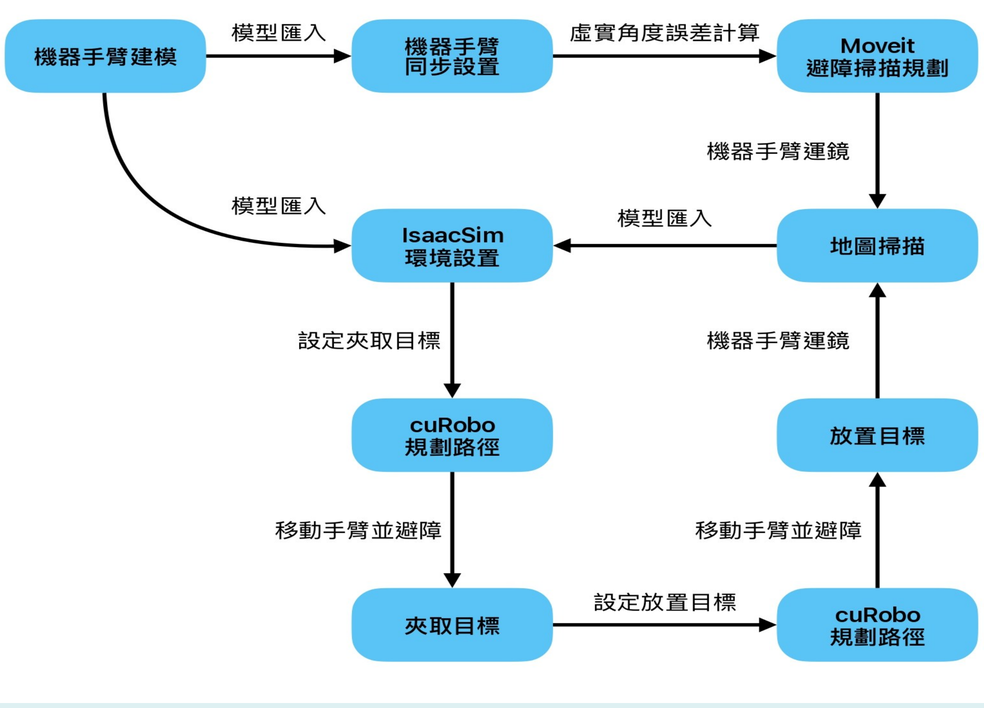
地圖建構與整合CUROBO之真實場域智慧型機械手臂多障礙避障控制

組別編號：　控制

摘要

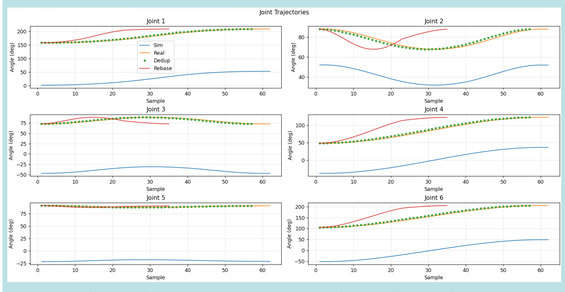
為了讓智慧型機械手臂從傳統工業拓展至醫療、居家及服務產業，本研究設計高擬真手臂建模、架設真實場域之方法與避障控制，進而讓機器手臂適應複雜多變的環境。我們使用FreeCAD建立手臂模型並設置Lula防碰撞球空間，整合MoveIt搭配Nvblox完成地圖掃描之粗略避障路徑規劃，其中透過Isaac Sim ROS2 Bridge連接Rviz與Nvidia IsaacSim進行雙模擬器軌跡驗證，以提升傳輸通訊品質。之後將地圖模型與手臂模型匯入IsaacSim，使用cuRobo演算法使手臂完成高精密的避障軌跡規劃之任務。

研究方法

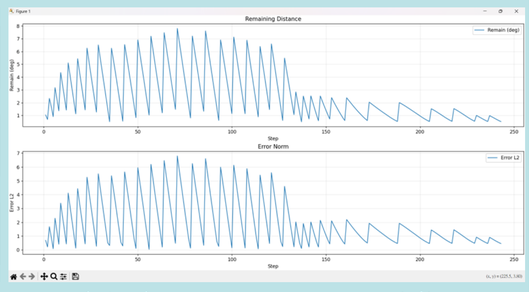


手臂控制

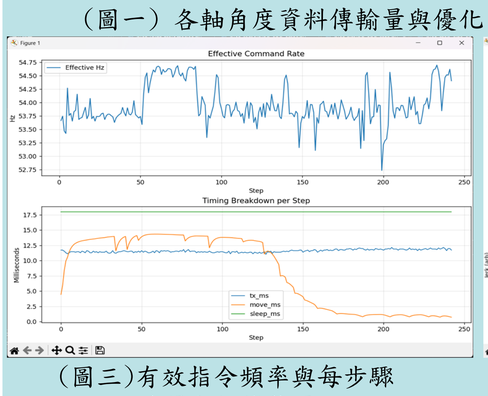
發現機器手臂移動卡頓不流暢，改良資料發佈前疏後密的形式並改用連續相對的G-code驅動命令，來完成傳輸端、接收端、驅動端同步，並採用EMA低通濾波去毛邊技術，壓制jerk的脈衝訊號，達成變加速度級別的穩定度



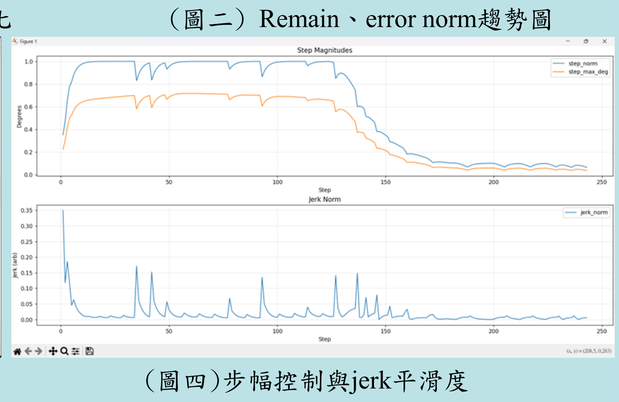
（圖一）各軸角度資料傳輸量與優化



（圖二）Remain、error norm趨勢圖



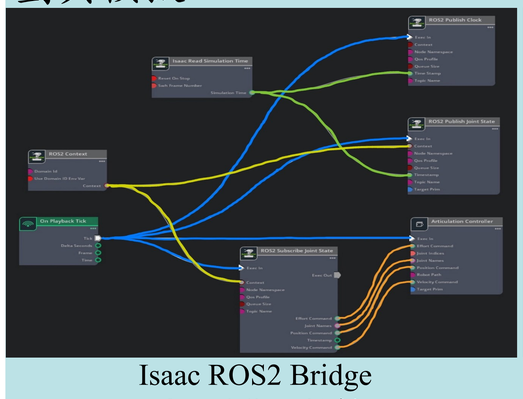
（圖三）有效指令頻率與每步驟時間分解趨勢圖



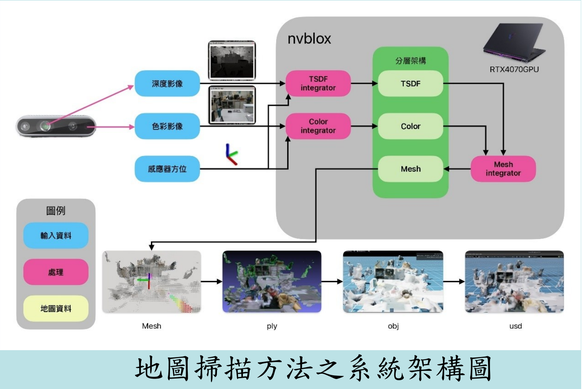
（圖四）步幅控制與jerk平滑度

研究產出

一、Isaac ROS2 Bridge細緻配置頻率、時間步長、關節角速度等命令，達成模擬器中機器手臂的運動軌跡高頻又密集的資料傳輸，緩解資料傳輸阻塞或爆頻等問題。
二、提出nvblox掃描真實場域建立 PLY、OBJ、USD轉檔供cuRobo與Isaac Sim，使掃描結果可真正進入後續避障規劃與模擬。



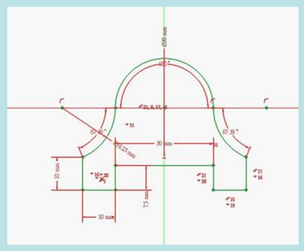
Isaac ROS2 Bridge
通訊效能之規劃



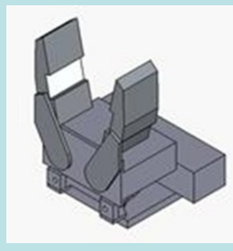
地圖掃描方法之系統架構圖

成果展示

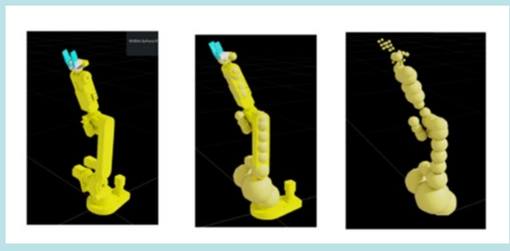
圖(A)~(H)展示從機器手臂建模與地圖掃描到模擬器同步連接真實機器手臂運作全過程。



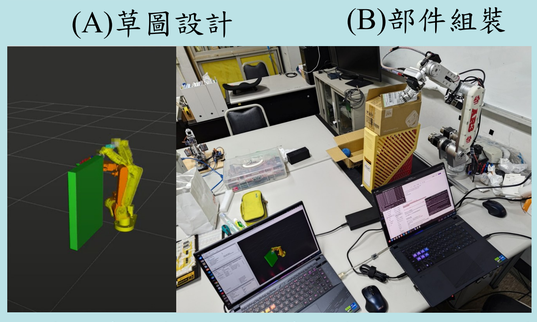
(A)草圖設計



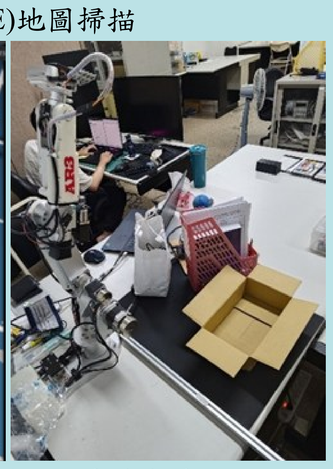
(B)部件組裝



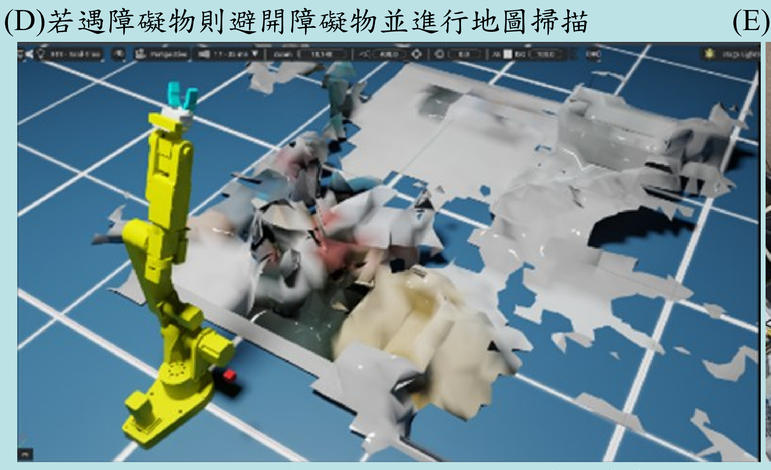
(C)Lula防碰撞球空間設置



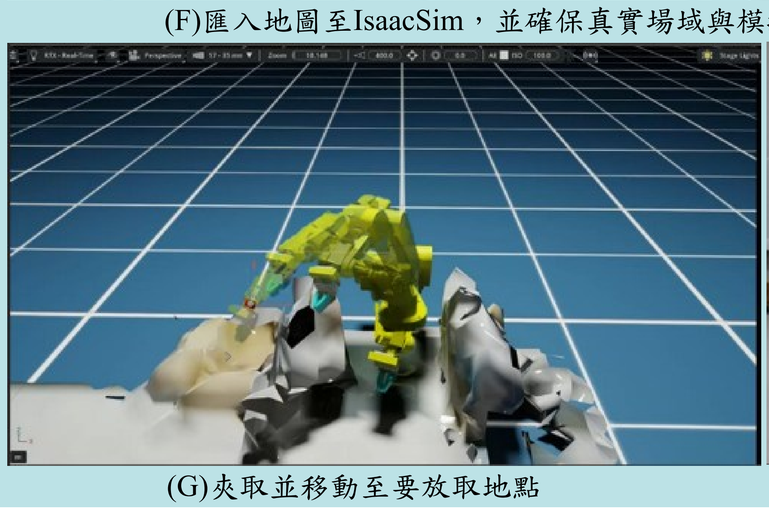
(D)若遇障礙物則避開障礙物並進行地圖掃描



(E)地圖掃描



(F)匯入地圖至IsaacSim，並確保真實場域與模擬器高度契合



(G)夾取並移動至要放取地點

(H)真實手臂避障控制

結論

本篇專題達成地圖掃描時的粗略避障功能與匯入地圖後的精密避障路徑規劃，並且整合實體機器手臂具有極佳的通訊效能與運行流暢的無碰撞平滑軌跡，是一個有研究價值和應用前景專題成果。

電機工程學系 / 指導老師:曾遠威 / 學生:許万峻、陳梓瑜

輕量化的基於流量分析與封包計數之V2N警示DDOS行為數位電路模擬、合成與驗證

組別編號:　　控制

摘要

本專題提出一套輕量、可合成的V2N（vehicle-to-network）DDoS行為警示電路。核心原理是在固定100ms視窗內，同步累計位元組數與封包數，並採雙層門檻與三視窗連續判定，分別輸出alert與severe_alert的DDoS警示訊號。電路以gate-level設計為主，有路徑短、面積小，易於時序收斂之特點。並以BSM-like的300B載荷之情境建立多個視窗流量的測試集，再於Quartus／Modelsim之開發環境中進行行為模擬、綜合與時序分析等功能正確性之驗證與探討。

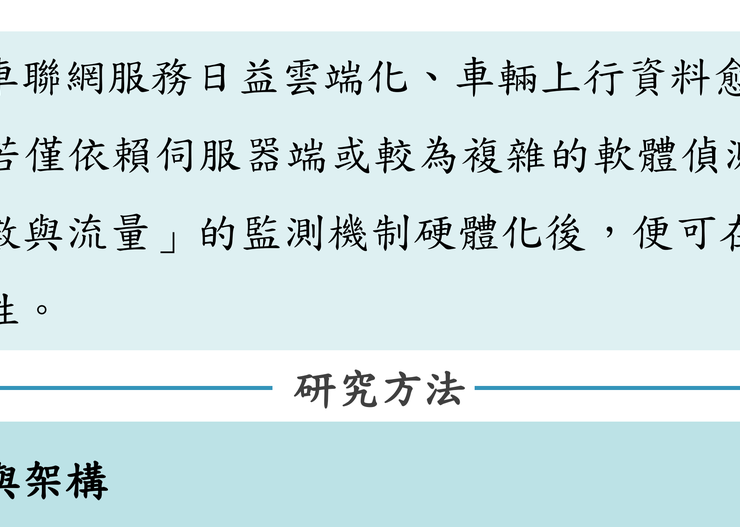
前言

隨著車聯網服務日益雲端化、車輛上行資料愈加頻繁，V2N 通道更容易出現壅塞，亦可能遭受 DDoS 以大量封包灌入而受干擾。若僅依賴伺服器端或較為複雜的軟體偵測，往往反應不夠即時，且對車載端算力負擔較高。為此，將「每 100 ms 統計封包數與流量」的監測機制硬體化後，便可在車端即時掌握速率異常，迅速發出告警或啟動限速，兼具低功耗、低延遲與高可靠性。

研究方法

電路階層與架構

本電路之架構規劃如下圖所示，將其依細部功能規劃，分別包含頂層、中層與底層模組。頂層為ddos_monitor_top；中層含 pulse_rate_counter、window_timer、cmp_ge、sync3_edge、latch_on_tick、streak3、alert_logic；底層含dffr/dffre、inc_n、eq_const，並各以專用Testbench驗證



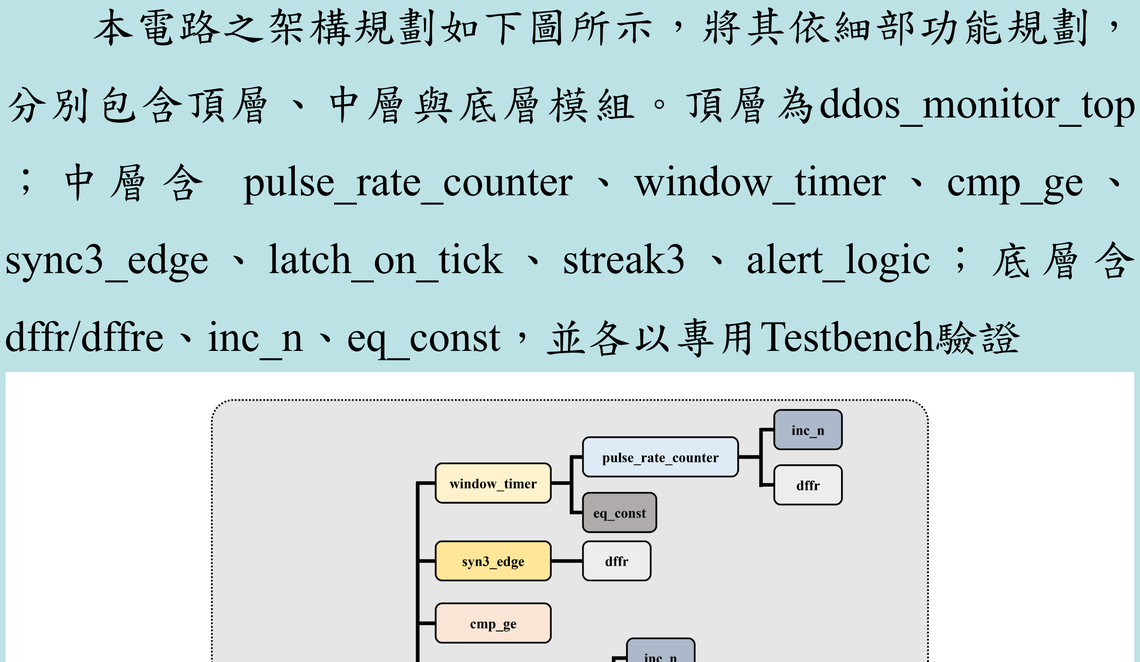


圖1、電路階層與系統架構圖

結論

本電路於頂層模組上以近扁平化方式設計，將演算法與其硬體實現，邏輯路徑短且穩定；window_timer 單一定時、四組門檻各以 cmp_ge 個別比較，bytes/packets 以成對模組構成，RTL以匯流排呈現計數與門檻，結構清晰、時序友善並具高度可重用性。整體行為以ModelSim驗證：TB 產生位元組/封包脈衝、依縮小視窗檢查 alert／severe_alert，涵蓋連續超門檻、重載、臨界與隨機百窗測試，確保在100 ms視窗與雙門檻、三視窗遲滯下皆能穩定運作。

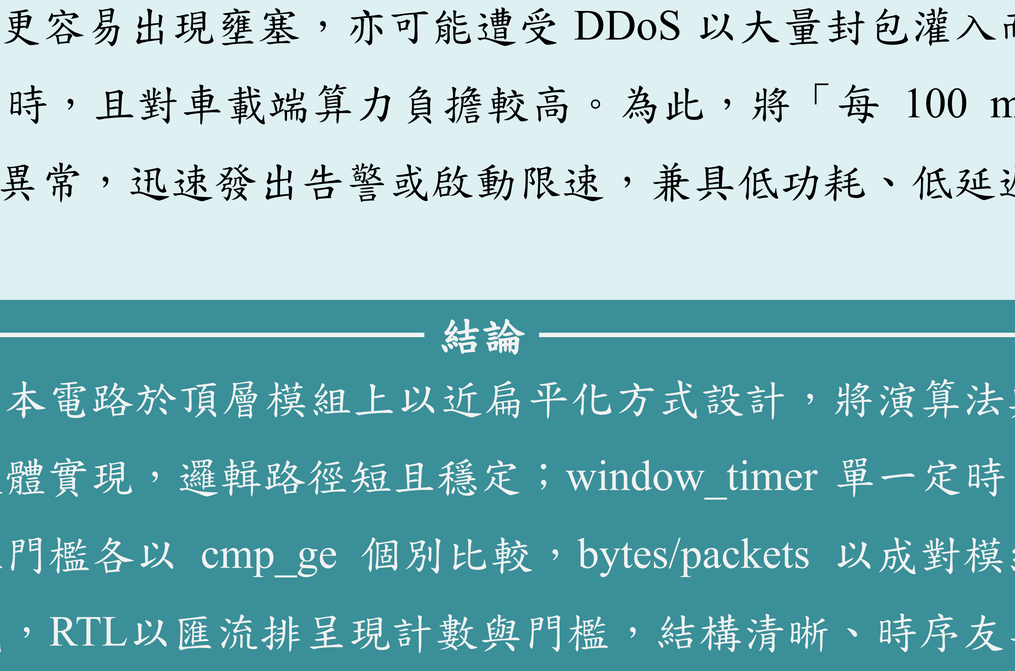
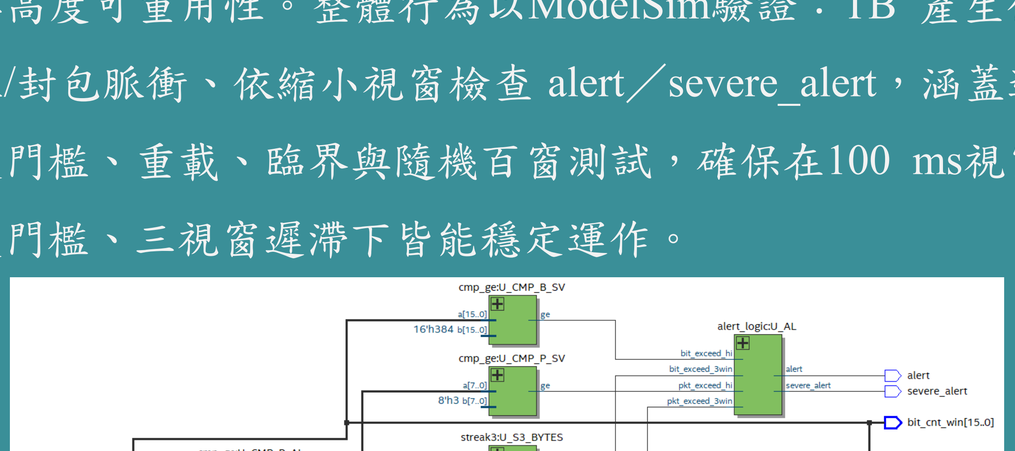


圖2、輕量化DDoS偵測警示電路之RTL圖



```
# [1] bytes=498 pkts=1  => alert=0 severe=0 (exp a=0 s=0)
# [2] bytes=402 pkts=2  => alert=0 severe=0 (exp a=0 s=0)
# [3] bytes=395 pkts=1  => alert=0 severe=0 (exp a=0 s=0)
# [4] bytes=499 pkts=1  => alert=0 severe=0 (exp a=0 s=0)
# [5] bytes=157 pkts=1  => alert=0 severe=0 (exp a=0 s=0)
# [6] bytes=0 pkts=0  => alert=0 severe=0 (exp a=0 s=0)
# [7] bytes=490 pkts=1  => alert=0 severe=0 (exp a=0 s=0)
# [8] bytes=410 pkts=2  => alert=0 severe=0 (exp a=0 s=0)
# [9] bytes=498 pkts=1  => alert=0 severe=0 (exp a=0 s=0)
# [10] bytes=448 pkts=2  => alert=0 severe=0 (exp a=0 s=0)
# [11] bytes=168 pkts=1  => alert=0 severe=0 (exp a=0 s=0)
# [12] bytes=465 pkts=2  => alert=0 severe=0 (exp a=0 s=0)
# [13] bytes=300 pkts=1  => alert=0 severe=0 (exp a=0 s=0)
# [14] bytes=418 pkts=2  => alert=0 severe=0 (exp a=0 s=0)
# [15] bytes=300 pkts=1  => alert=0 severe=0 (exp a=0 s=0)
# [16] bytes=370 pkts=2  => alert=0 severe=0 (exp a=0 s=0)
# [17] bytes=427 pkts=1  => alert=0 severe=0 (exp a=0 s=0)
# [18] bytes=497 pkts=2  => alert=0 severe=0 (exp a=0 s=0)
# [19] bytes=275 pkts=1  => alert=0 severe=0 (exp a=0 s=0)
# [20] bytes=426 pkts=2  => alert=0 severe=0 (exp a=0 s=0)
# [21] bytes=300 pkts=1  => alert=0 severe=0 (exp a=0 s=0)
# [22] bytes=378 pkts=2  => alert=0 severe=0 (exp a=0 s=0)
# [23] bytes=481 pkts=2  => alert=0 severe=0 (exp a=0 s=0)
# [24] bytes=300 pkts=1  => alert=0 severe=0 (exp a=0 s=0)
# [25] bytes=499 pkts=1  => alert=0 severe=0 (exp a=0 s=0)
# [26] bytes=85 pkts=1  => alert=0 severe=0 (exp a=0 s=0)
# [27] bytes=300 pkts=1  => alert=0 severe=0 (exp a=0 s=0)
# [28] bytes=497 pkts=2  => alert=0 severe=0 (exp a=0 s=0)
# [29] bytes=497 pkts=2  => alert=0 severe=0 (exp a=0 s=0)
# [30] bytes=440 pkts=2  => alert=0 severe=0 (exp a=1 s=0)
#   MISMATCH alert at window 30!
# [31] bytes=497 pkts=2  => alert=1 severe=0 (exp a=1 s=0)
# [32] bytes=243 pkts=1  => alert=0 severe=0 (exp a=0 s=0)
# [33] bytes=394 pkts=2  => alert=0 severe=0 (exp a=0 s=0)
# [34] bytes=0 pkts=0  => alert=0 severe=0 (exp a=0 s=0)
# [35] bytes=0 pkts=0  => alert=0 severe=0 (exp a=0 s=0)
# [36] bytes=497 pkts=2  => alert=0 severe=0 (exp a=0 s=0)
# [37] bytes=300 pkts=1  => alert=0 severe=0 (exp a=0 s=0)
# [38] bytes=0 pkts=0  => alert=0 severe=0 (exp a=0 s=0)
# [39] bytes=300 pkts=0  => alert=0 severe=0 (exp a=0 s=0)
# [40] bytes=498 pkts=2  => alert=0 severe=0 (exp a=0 s=0)
# [41] bytes=403 pkts=2  => alert=0 severe=0 (exp a=0 s=0)
# [42] bytes=300 pkts=1  => alert=0 severe=0 (exp a=0 s=0)
# [43] bytes=497 pkts=2  => alert=0 severe=0 (exp a=0 s=0)
# [44] bytes=497 pkts=2  => alert=0 severe=0 (exp a=0 s=0)
# [45] bytes=409 pkts=2  => alert=0 severe=0 (exp a=1 s=0)
#   MISMATCH alert at window 45!
# [46] bytes=410 pkts=2  => alert=1 severe=0 (exp a=1 s=0)
# [47] bytes=499 pkts=1  => alert=0 severe=0 (exp a=0 s=0)
# [48] bytes=314 pkts=2  => alert=0 severe=0 (exp a=0 s=0)
# [49] bytes=315 pkts=2  => alert=0 severe=0 (exp a=0 s=0)
# [50] bytes=0 pkts=0  => alert=0 severe=0 (exp a=0 s=0)
# [51] bytes=300 pkts=1  => alert=0 severe=0 (exp a=0 s=0)
# [52] bytes=300 pkts=1  => alert=0 severe=0 (exp a=0 s=0)
# [53] bytes=419 pkts=2  => alert=0 severe=0 (exp a=0 s=0)
# [54] bytes=498 pkts=1  => alert=0 severe=0 (exp a=0 s=0)
# [55] bytes=322 pkts=2  => alert=0 severe=0 (exp a=0 s=0)
# [56] bytes=276 pkts=0  => alert=0 severe=0 (exp a=0 s=0)
# [57] bytes=23 pkts=1  => alert=0 severe=0 (exp a=0 s=0)
# [58] bytes=0 pkts=0  => alert=0 severe=0 (exp a=0 s=0)
# [59] bytes=175 pkts=1  => alert=0 severe=0 (exp a=0 s=0)
# [60] bytes=300 pkts=1  => alert=0 severe=0 (exp a=0 s=0)
# [61] bytes=321 pkts=1  => alert=0 severe=0 (exp a=0 s=0)
# [62] bytes=279 pkts=1  => alert=0 severe=0 (exp a=0 s=0)
# [63] bytes=369 pkts=1  => alert=0 severe=0 (exp a=0 s=0)
# [64] bytes=498 pkts=1  => alert=0 severe=0 (exp a=0 s=0)
# [65] bytes=299 pkts=1  => alert=0 severe=0 (exp a=0 s=0)
# [66] bytes=33 pkts=1  => alert=0 severe=0 (exp a=0 s=0)
# [67] bytes=484 pkts=2  => alert=0 severe=0 (exp a=0 s=0)
# [68] bytes=0 pkts=0  => alert=0 severe=0 (exp a=0 s=0)
# [69] bytes=0 pkts=0  => alert=0 severe=0 (exp a=0 s=0)
# [70] bytes=499 pkts=1  => alert=0 severe=0 (exp a=0 s=0)
# [70] bytes=499 pkts=1  => alert=0 severe=0 (exp a=0 s=0)
# [71] bytes=449 pkts=2  => alert=0 severe=0 (exp a=0 s=0)
# [72] bytes=498 pkts=1  => alert=0 severe=0 (exp a=0 s=0)
# [73] bytes=342 pkts=2  => alert=0 severe=0 (exp a=0 s=0)
# [74] bytes=300 pkts=1  => alert=0 severe=0 (exp a=0 s=0)
# [75] bytes=498 pkts=1  => alert=0 severe=0 (exp a=0 s=0)
# [76] bytes=448 pkts=2  => alert=0 severe=0 (exp a=0 s=0)
# [77] bytes=247 pkts=1  => alert=0 severe=0 (exp a=0 s=0)
# [78] bytes=498 pkts=1  => alert=0 severe=0 (exp a=0 s=0)
# [79] bytes=498 pkts=2  => alert=0 severe=0 (exp a=0 s=0)
# [80] bytes=152 pkts=1  => alert=0 severe=0 (exp a=0 s=0)
# [81] bytes=0 pkts=0  => alert=0 severe=0 (exp a=0 s=0)
# [82] bytes=297 pkts=0  => alert=0 severe=0 (exp a=0 s=0)
# [83] bytes=154 pkts=1  => alert=0 severe=0 (exp a=0 s=0)
# [84] bytes=494 pkts=1  => alert=0 severe=0 (exp a=0 s=0)
# [85] bytes=499 pkts=1  => alert=0 severe=0 (exp a=0 s=0)
# [86] bytes=58 pkts=1  => alert=0 severe=0 (exp a=0 s=0)
# [87] bytes=359 pkts=2  => alert=0 severe=0 (exp a=0 s=0)
# [88] bytes=438 pkts=1  => alert=0 severe=0 (exp a=0 s=0)
# [89] bytes=161 pkts=1  => alert=0 severe=0 (exp a=0 s=0)
# [90] bytes=462 pkts=2  => alert=0 severe=0 (exp a=0 s=0)
# [91] bytes=335 pkts=1  => alert=0 severe=0 (exp a=0 s=0)
# [92] bytes=265 pkts=1  => alert=0 severe=0 (exp a=0 s=0)
# [93] bytes=498 pkts=1  => alert=0 severe=0 (exp a=0 s=0)
# [94] bytes=318 pkts=2  => alert=0 severe=0 (exp a=0 s=0)
# [95] bytes=319 pkts=2  => alert=0 severe=0 (exp a=0 s=0)
# [96] bytes=0 pkts=0  => alert=0 severe=0 (exp a=0 s=0)
# [97] bytes=300 pkts=1  => alert=0 severe=0 (exp a=0 s=0)
# [98] bytes=300 pkts=1  => alert=0 severe=0 (exp a=0 s=0)
# [99] bytes=176 pkts=0  => alert=0 severe=0 (exp a=0 s=0)
# [100] bytes=498 pkts=1  => alert=0 severe=0 (exp a=0 s=0)
# TEST FAILED with 2 error(s).
```

圖3、輕量化DDoS偵測警示電路之驗證回報